Rockchip RK2118 RKNN SDK Quick Start

文件标识：RK-JC-YF-414

发布版本：V2.4.0

日期：2026-01-17

文件密级：□绝密 □秘密 □内部资料 ■公开

免责声明

本文档按"现状"提供，瑞芯微电子股份有限公司（"本公司"，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

"Rockchip"、"瑞芯微"、"瑞芯"均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

版权所有 © 2026 瑞芯微电子股份有限公司

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文介绍如何快速上手使用RK2118 RKNN SDK。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明	核定人
V2.1.0	HPC	2024-7-30	支持RK2118	熊伟
v2.2.0	HPC	2024-09-04	更新版本号	熊伟
v2.3.0	HPC	2024-11-06	增加通过 pip 源安装 RKNN-Toolkit2 的支持	熊伟
v2.3.2	HPC	2025-04-03	更新版本号	熊伟
v2.4.0	HPC	2026-01-17	更新版本号	熊伟

目录

Rockchip RK2118 RKNN SDK Quick Start

- 1 主要说明
- 2 准备开发板
  - 2.1 开发板和连接工具介绍
  - 2.2 连接开发板
- 3 准备开发环境
  - 3.1 下载 RKNN 相关仓库
  - 3.2 安装 RKNN-Toolkit2 环境
    - 3.2.1 安装 Python
      - 3.2.1.1 安装 miniforge
      - 3.2.1.2 使用 miniforge 创建 Python 环境
    - 3.2.2 安装 RKNN-Toolkit2
    - 3.2.3 验证是否安装成功
  - 3.3 准备RK2118 SDK
    - 3.3.1 工程下载
    - 3.3.2 编译环境准备
    - 3.3.3 RK2118固件编译
      - 3.3.3.1 开启RKNPU支持
      - 3.3.3.2 编译固件及打包
      - 3.3.3.3 烧写固件
- 4 运行示例程序
  - 4.1 RKNN SDK 介绍
  - 4.2 模型转换
  - 4.3 RKNN C Demo 使用方法
    - 4.3.1 编译rknn_benchmark
    - 4.3.2 推送文件到板端
    - 4.3.3 板端运行 Demo
- 5 Docker 中进行 RKNN模型转换（可选）
  - 5.1 安装 Docker
  - 5.2 在 Docker 中安装 RKNN-Toolkit2 环境
    - 5.2.1 准备 RKNN-Toolkit2 镜像
    - 5.2.2 查询镜像信息
  - 5.3 RKNN 模型转换
- 6 常见问题
  - 6.1 命令 adb devices 查看不到设备
  - 6.2 rknn_server 服务的作用是什么?

7 参考文档

1 主要说明

此文档面向零基础用户详细介绍如何快速在计算机上使用 RKNN-Toolkit2 完成模型转换，并通过 RKNPU2 部署到 Rockchip 开发板上。

支持的平台：RK2118系列。

2 准备开发板

本章将介绍如何将开发板连接到计算机。分为两个部分：

- 开发板和连接工具介绍
- 连接开发板

2.1 开发板和连接工具介绍

1. 开发板

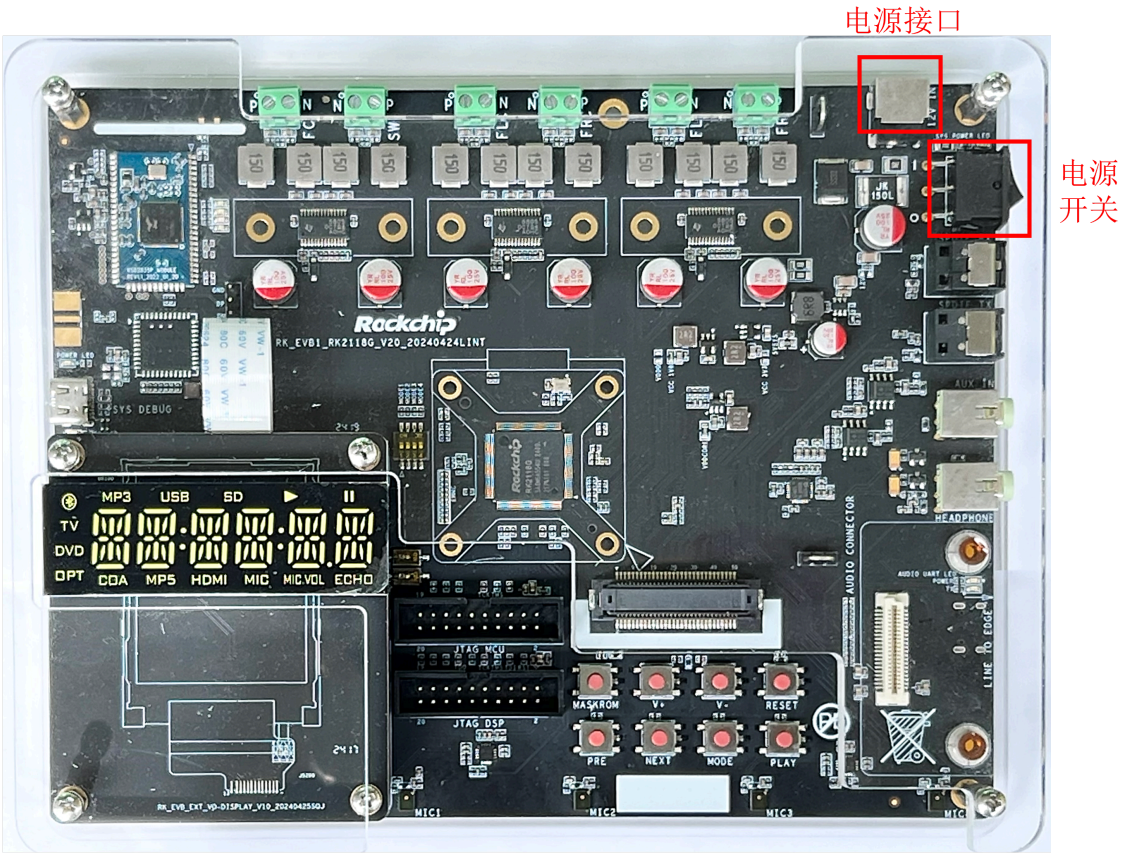

图2-1 RK2118开发板

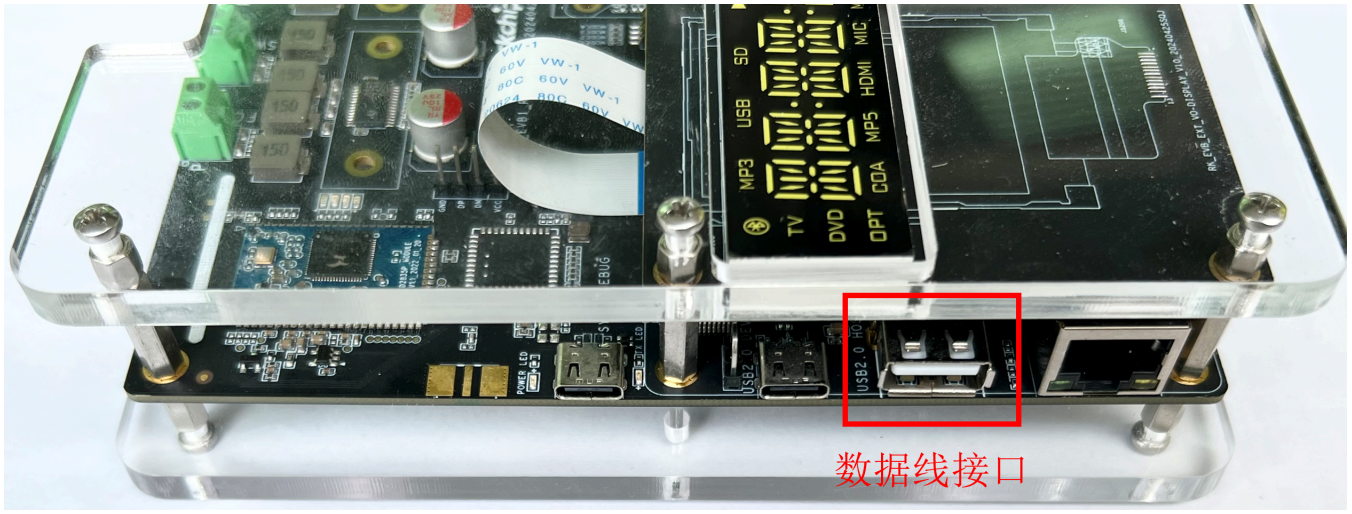

图2-2 RK2118开发板侧面

2. 连接开发板和计算机的数据线

图2-3 USB-C 数据线

图2-4 USB-A 数据线

3. 电源适配器

图2-5 输出 12V-2A 的电源适配器

2.2 连接开发板

下面以 RK2118 为例说明如何将开发板连至计算机：

1. 准备一台操作系统为 Ubuntu18.04 / Ubuntu20.04 / Ubuntu22.04 的计算机。
2. 找到图2-1中电源接口的位置，连接开发板电源适配器。

3. 使用数据线连接开发板与计算机（注意，由于生产批次不同，开发板的数据线接口类型和位置可能会发生变化。通常情况下，开发板上印有 SYSTEM DEBUG 字样的接口为串口，USB2.0 HOST为烧录口）。
4. 打开电源开关，等待开发板系统启动完成。
5. 查看开发板是否连接至计算机

使用TYPEC线连接开发板的"SYSTEM DEBUG"口和计算机。在计算机上使用串口工具查看RK2118的情况。比如使用minicom。串口波特率为

```
1500000 8N1
```

3 准备开发环境

本章介绍如何在计算机中直接安装开发环境，后续的示例程序运行过程也将以直接安装为例说明。如果需要在 Docker 环境中运行示例程序，可以参考第5章中的内容准备开发环境。

本章分为三个部分：

- 下载 RKNN 相关仓库
- 安装RKNN-Toolkit2 环境
- 准备RK2118 SDK

3.1 下载 RKNN 相关仓库

建议新建一个目录用来存放 RKNN 仓库，例如新建一个名称为 Projects 的文件夹，并将 RKNN-Toolkit2仓库存放至该目录下，参考命令如下：

```
# 新建 Projects 文件夹
mkdir Projects

# 进入该目录
cd Projects

# 下载 RKNN-Toolkit2 仓库
git clone https://github.com/airockchip/rknn-toolkit2.git --depth 1
# 注意：
# 1.参数 --depth 1 表示只克隆最近一次 commit
# 2.如果遇到 git clone 失败的情况，也可以直接在 github 中下载压缩包到本地，然后解压至该目录
```

整体目录结构如下：

```
Projects
├── rknn-toolkit2
│   ├── doc
│   ├── rknn-toolkit2
│   │   ├── packages
│   │   ├── docker
│   │   └── ...
│   ├── rknpu2
│   │   ├── runtime
│   │   └── ...
│   └── ...
```

3.2 安装 RKNN-Toolkit2 环境

3.2.1 安装 Python

如果系统中没有安装 Python 3.8（建议版本），或者同时有多个版本的 Python 环境，建议使用 miniforge 创建新的 Python 3.8 环境。

3.2.1.1 安装 miniforge

在计算机的终端窗口中执行以下命令，检查是否安装 miniforge，若已安装则可省略此节步骤。

```
conda -V
# 参考输出信息：conda 24.3.0 ，表示 conda 版本为 24.3.0
# 如果提示 conda: command not found，则表示未安装 miniforge
```

如果没有安装 miniforge，可以通过下面的链接下载 miniforge 安装包：

```
wget -c https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh
```

然后通过以下命令安装 miniforge：

```
chmod 777 Miniforge3-Linux-x86_64.sh
bash Miniforge3-Linux-x86_64.sh
```

3.2.1.2 使用 miniforge 创建 Python 环境

在计算机的终端窗口中，执行以下命令进入 Conda base 环境：

```
source /opt/miniforge3/bin/activate # miniforge3 安装的目录
# 成功后，命令行提示符会变成以下形式：
# (base) xxx@xxx:~$
```

通过以下命令创建名称为 toolkit2 的 Python 3.8 环境：

```
conda create -n toolkit2 python=3.8
```

激活 toolkit2 环境，后续将在此环境中安装 RKNN-Toolkit2：

```
conda activate toolkit2
# 成功后，命令行提示符会变成以下形式：
# (toolkit2) xxx@xxx:~$
```

3.2.2 安装 RKNN-Toolkit2

激活 toolkit2 环境后，可通过 pip 源或者本地 wheel 包这两种方式安装 RKNN-Toolkit2：

- 通过 pip 源安装

```
pip install rknn-toolkit2 -i https://pypi.org/simple

# 如果已安装 RKNN-Toolkit2，可通过以下命令升级 RKNN-Toolkit2
pip install rknn-toolkit2 -i https://pypi.org/simple --upgrade
```

- 通过本地 wheel 包安装

```
# 进入 rknn-toolkit2 目录
cd Projects/rknn-toolkit2/rknn-toolkit2

# 请根据不同的 python 版本及处理器架构，选择不同的 requirements 文件：
# 其中 cpxx 是 python 版本号
pip install -r packages/x86_64/requirements_cpxx.txt
# pip install -r packages/arm64/arm64_requirements_cpxx.txt

# 安装 RKNN-Toolkit2
# 请根据不同的 python 版本及处理器架构，选择不同的 wheel 安装包文件：
# 其中 x.x.x 是 RKNN-Toolkit2 版本号，cpxx 是 python 版本号
pip install packages/x86_64/rknn_toolkit2-x.x.x-cpxx-cpxx-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl
# pip install packages/arm64/rknn_toolkit2-x.x.x-cpxx-cpxx-
manylinux_2_17_aarch64.manylinux2014_aarch64.whl
```

3.2.3 验证是否安装成功

执行以下命令，若没有报错，则代表 RKNN-Toolkit2 环境安装成功。

```
# 进入 Python 交互模式
python

# 导入 RKNN 类
from rknn.api import RKNN
```

如果安装失败，请查阅 《Rockchip_RKNPU_User_Guide_RKNN_SDK_V2.1.0_CN.pdf》文档中的第 10.2 章节 “工具安装问题”，其中详细介绍了 RKNN-Toolkit2 环境安装失败的解决方法。

3.3 准备RK2118 SDK

NPU的相关示例包含在RK2118 SDK中，因此需要先准备RK2118的SDK源码。

3.3.1 工程下载

- repo 工具下载，如果已经安装了 repo ，请略过该步骤。

```
git clone ssh://git@www.rockchip.com.cn/repo/rk/tools/repo -b stable
export PATH=/path/to/repo:$PATH
```

- 工程下载

```
mkdir rt-thread
cd rt-thread
repo init --repo-url ssh://git@www.rockchip.com.cn/repo/rk/tools/repo -u
ssh://git@www.rockchip.com.cn/rtos/rt-thread/rk/platform/release/manifests -b master -m
rk2118.xml
.repo/repo/repo sync
```

注意：该工程访问需要权限，请找RK人员申请。

3.3.2 编译环境准备

请参考"rt-thread/RKDocs/manuals/rk2118"目录下的《Rockchip_Developer_Guide_RK2118.pdf》搭建开发环境。

3.3.3 RK2118固件编译

3.3.3.1 开启RKNPU支持

参考《Rockchip_Developer_Guide_RK2118.pdf》第4章节"CPU固件编译"。在rt-thread目录下执行如下语句：

```
cd bsp/rockchip/rk2118
cp board/evb/defconfig .config
scons --menuconfig
```

请确保如下选项已经打开：

```
RT-Thread Components  --->
  [*] DFS: device virtual file system  --->
       [*]   Using posix-like functions, open/read/write/close
       [*]   Using working directory
       (4)   The maximal number of mounted file system
       (4)   The maximal number of file system type
       (16)  The maximal number of opened files
       [*]   Using mount table for file system

       [*]   Enable elm-chan fatfs
RT-Thread rockchip RK2118 drivers  --->
  [*] Enable RKNPU
```

3.3.3.2 编译固件及打包

```
cd bsp/rockchip/rk2118
scons -j32
./mkimage.sh board/evb/setting.ini
```

编译完成后将生成Image/Firmware.img等文件

3.3.3.3 烧写固件

参考《Rockchip_Developer_Guide_RK2118》“4.1.3.4 固件烧录”章节。

4 运行示例程序

本章将介绍如何快速在开发板上运行示例程序，内容分为三个部分：

- RKNN SDK介绍
- RKNN 模型转换
- RKNN C Demo 使用方法

4.1 RKNN SDK 介绍

RK2118 RKNN SDK使用与RV1103/RV1106 SDK类似的mini runtime接口。该SDK存放在rt-thread/components/rknpu目录中，其目录结构如下：

```
rknpu
├── examples     # 示例代码
├── include      # 头文件
├── lib          # 运行时库
├── rknn_server  # 与DSP交互服务程序
└── SConscript
```

其中，examples 目录包括了一些使用示例，例如 MobileNet 等。以rknn_benchmark为例，其目录结构如下：

```
rknn_benchmark/
├── README.md
├── res
│   ├── convert.py          #模型转换脚本
├── SConscript              #编译脚本
├── src
│   └── rknn_benchmark.c    #测试程序
└── tools
    └── update_res.py       #生成模型输入文件
```

4.2 模型转换

下面以 MobileNetV1为例，介绍如何转换RKNN 模型。

进入 rt-thread/components/rknpu/examples/rknn_benchmark/res 目录，运行 convert.py 脚本，该脚本将原始的pb模型转成 RKNN 模型，参考命令如下：

```
cd rt-thread/components/rknpu/examples/rknn_benchmark/res
python convert.py
```

运行成功将生成mobilenetv1.rknn及dog.bin。dog.bin为模型输入文件。

4.3 RKNN C Demo 使用方法

下面以 rknn_benchmark为例，介绍 RKNN C Demo 的使用方法。

注：不同的 RKNN C Demo 用法存在差异，请按照各自目录下 README.md 中的步骤运行。

4.3.1 编译rknn_benchmark

参考第3.3.3 RK2118固件编译章节编译的固件已经默认包含rknn_benchmark程序，可以在系统启动后，直接运行rknn_benchmark。

4.3.2 推送文件到板端

由于板子的存储空间有限，在测试时，可以使用U盘存储RKNN模型（只支持FAT32格式），实际使用时，可以将RKNN模型打包到固件中。这里以U盘为例。将"4.2 模型转换"生成的mobilenetv1.rknn及dog.bin拷贝到U盘根目录。并将U盘插入到板子的"USB 2.0 HOST"口。在板子执行如下命令可以看到模型文件。

```
ls /
```

4.3.3 板端运行 Demo

执行以下命令，在板端运行rknn_benchmark：

```
# 进入板端（串口）

cd /

# 运行可执行文件
# 用法: ./rknn_benchmark <model_path> <input_path>
rknn_benchmark mobilenetv1.rknn dog.bin
```

运行结果参考

```
[156] 19.187500
[155] 16.234375
[205] 13.539062
[284] 12.226562
[260] 10.179687
```

注：运行结果只打印TOP5，默认没有做softmax。

5 Docker 中进行 RKNN模型转换（可选）

如果需要在 Docker 环境中运行 RKNN Python Demo，可以参考本章中的内容准备开发环境。

请特别注意，这里提供了一个包含 RKNN-Toolkit2 环境的 Docker 镜像，允许用户在其中直接运行 RKNN Python Demo 而无需担心环境安装问题。但是，该 Docker 镜像中只包含纯净的 RKNN-Toolkit2 环境，仅适用于运行 RKNN Python Demo。

此章内容分为三个部分：

- 安装 Docker
- 在 Docker 中安装 RKNN-Toolkit2 环境
- RKNN 模型转换

5.1 安装 Docker

如果已安装 Docker ，可跳过此步骤。如果没有安装，请根据官方手册进行安装。

Docker 安装官方手册链接：https://docs.docker.com/install/linux/docker-ce/ubuntu/

注意事项：需要将用户添加到 docker 用户组。

```
# 创建docker用户组
sudo groupadd docker
# 把当前用户加入docker用户组
sudo usermod -aG docker $USER
# 更新激活docker用户组
newgrp docker

# 验证不需要sudo执行docker命令
docker run hello-world
```

成功安装的参考输出信息如下：

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:88ec0acaa3ec199d3b7eaf73588f4518c25f9d34f58ce9a0df68429c5af48e8d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
```

5.2 在 Docker 中安装 RKNN-Toolkit2 环境

5.2.1 准备 RKNN-Toolkit2 镜像

本节介绍两种创建 RKNN-Toolkit2 镜像环境的方式，可任选一种方式进行创建。

1. 通过 Dockerfile 创建 RKNN Toolkit2 镜像

   在 RKNN-Toolkit2 工程中 docker/docker_file 文件夹下，提供了构建 RKNN-Toolkit2 开发环境的 Dockerfile 文件，用户通过 docker build命令创建镜像，如下所示：

```
# 注：以下 xx 和 x.x.x 代表版本号，请根据实际数值进行替换
cd Projects/rknn-toolkit2/rknn-toolkit2/docker/docker_file/ubuntu_xx_xx_cpxx

docker build -f Dockerfile_ubuntu_xx_xx_for_cpxx -t rknn-toolkit2:x.x.x-cpxx .
```

2. 通过加载已打包的 Docker 镜像文件创建 RKNN Toolkit2 镜像

   通过如下链接下载对应版本的 RKNN-Toolkit2 工程文件，解压后在 docker/docker_image 文件夹下提供了已打包所有开发环境的 Docker 镜像。

   Docker 镜像文件网盘下载链接：https://console.zbox.filez.com/l/I00fc3 （提取码：rknn）

   执行以下命令加载对应 Python 版本的镜像文件。

```
# 注：x.x.x 代表 RKNN-Toolkit2 的版本号，cpxx 代表的是 Python 的版本
docker load --input rknn-toolkit2-x.x.x-cpxx-docker.tar.gz
```

5.2.2 查询镜像信息

创建或加载镜像成功后，可以通过以下命令查看 docker 的镜像信息。

```
docker images
```

相应的 RKNN-Toolkit2 镜像信息显示如下：

```
REPOSITORY        TAG             IMAGE ID        CREATED        SIZE
rknn-toolkit2     x.x.x-cpxx      xxxxxxxxxxxx    1 hours ago    5.89GB
```

5.3 RKNN 模型转换

- 映射文件，并运行容器

  请参考第3.1章节在本地下载好 RT-Thread 工程，然后将其映射进容器中，并通过 docker run 命令运行容器。运行后将进入容器的 bash 环境。参考命令如下：

```
# 使用 docker run 命令创建并运行 RKNN Toolkit2 容器
# 并通过附加 -v <host src folder>:<image dst folder> 参数，将本地文件映射进容器中
docker run -t -i --privileged \
    -v /dev/bus/usb:/dev/bus/usb \
    -v /Projects/rt-thread:/rt-thread \
    rknn-toolkit2:x.x.x-cpxx \
    /bin/bash
```

- 模型转换

  进入rt-thread/components/rknpu/examples/rknn_benchmark/res 目录转换模型，参考命令如下：

```
cd rt-thread/components/rknpu/examples/rknn_benchmark/res
python convert.py
```

6 常见问题

6.1 命令 adb devices 查看不到设备

目前，RK2118不支持通过adb调试，因此没有adb设备

6.2 rknn_server 服务的作用是什么?

RK2118中的rknn_server并不是用来做RKNN-Toolkit2 的连板调试功能，而是用于DSP与MCU进行交互的服务。

7 参考文档

- 有关RKNN-Toolkit2和RKNPU2更详细的用法和接口说明，请参考《Rockchip_RKNPU_User_Guide_RKNN_SDK_V2.1.0_CN.pdf》手册。
- RK2118的相关文档请参考《Rockchip_Developer_Guide_RK2118.pdf》